import java.util.Scanner;

import java.util.Arrays;

import java.util.Comparator;

public class FractionalKnapSack {

// Item value class

static class ItemValue {

int profit, weight;

public ItemValue(int val, int wt) {

this.weight = wt;

this.profit = val;

}

}

// Function to get maximum value

private static double getMaxValue(ItemValue[] arr, int capacity) {

Arrays.sort(arr, new Comparator<ItemValue>() {

@Override

public int compare(ItemValue item1, ItemValue item2) {

double cpr1 = (double) item1.profit / (double) item1.weight;

double cpr2 = (double) item2.profit / (double) item2.weight;

if (cpr1 < cpr2)

return 1;

else

return -1;

}

});

double totalValue = 0d;

for (ItemValue i : arr) {

int curWt = (int) i.weight;

int curVal = (int) i.profit;

if (capacity - curWt >= 0) {

// This weight can be picked whole

capacity = capacity - curWt;

totalValue += curVal;

} else {

// Item can't be picked whole

double fraction = (double) capacity / (double) curWt;

totalValue += (curVal \* fraction);

capacity = (int) (capacity - (curWt \* fraction));

break;

}

}

return totalValue;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Enter the knapsack capacity: ");

int capacity = scanner.nextInt();

System.out.print("Enter the number of items: ");

int n = scanner.nextInt();

ItemValue[] arr = new ItemValue[n];

for (int i = 0; i < n; i++) {

System.out.print("Enter profit for item " + (i + 1) + ": ");

int profit = scanner.nextInt();

System.out.print("Enter weight for item " + (i + 1) + ": ");

int weight = scanner.nextInt();

arr[i] = new ItemValue(profit, weight);

}

double maxValue = getMaxValue(arr, capacity);

System.out.println("Maximum value in the knapsack: " + maxValue);

scanner.close();

}

}

Enter the knapsack capacity: 50

Enter the number of items: 3

Enter profit for item 1: 60

Enter weight for item 1: 10

Enter profit for item 2: 100

Enter weight for item 2: 20

Enter profit for item 3: 120

Enter weight for item 3: 30

Maximum value in the knapsack: 240.0